

Comparator Datasheet CMP V 1.00

Copyright © 2013 Cypress Semiconductor Corporation. All Rights Reserved.

| UM Configurations | PSoC® Resources | | | API Memory (Bytes) | | Pins (per External I/O) |
|--|-----------------|----------------|------------------|--------------------|-----|-------------------------|
| | CMP | LPF (Optional) | Analog Interrupt | Flash | ROM | |
| CY8C24x93 | | | | | | |
| Without LPF and ISR | 1 | 0 | 0 | 106 | 0 | 0 to 3* |
| With LPF only** | 1 | 1 | 0 | 124 | 0 | 0 to 3* |
| With ISR only | 1 | 0 | 1 | 125 | 0 | 0 to 3* |
| With LPF and ISR** | 1 | 1 | 1 | 143 | 0 | 0 to 3* |
| * Depends on UM parameters. | | | | | | |
| ** These configurations are available only for CY8C24193 and CY8C24493 devices | | | | | | |

For one or more fully configured, functional code examples that use this user module go to www.cypress.com/psocexampleprojects.

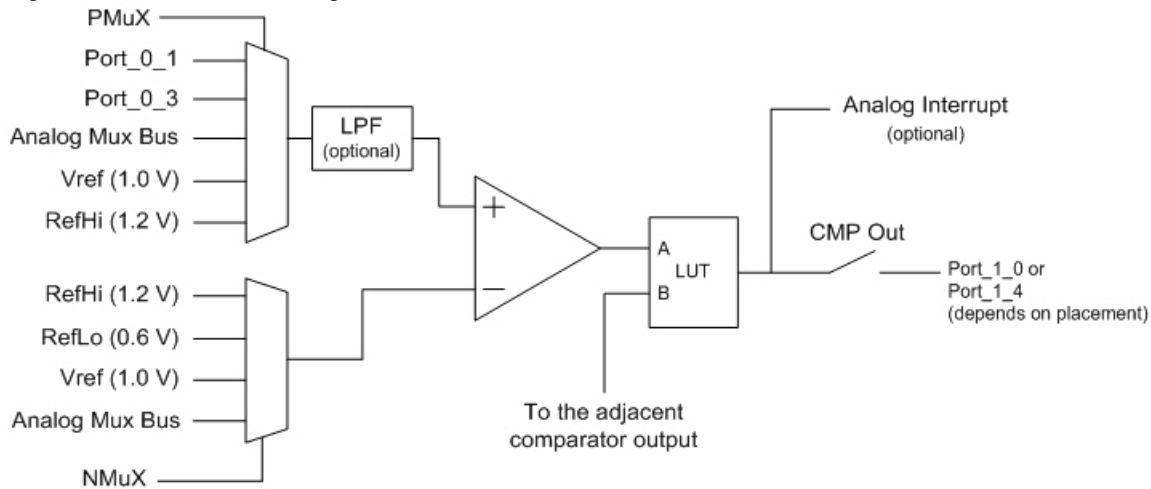
Features and Overview

- Provides two comparators separately
- LPF is accessible to only one instance through the user module placement; however, you can modify the code and use it for both instances
- Supports AMUX connection in both the inputs
- Supports routing CMP output to an external pin

Functional Description

The CMP User Module offers a simple comparator. You can route a pin to the positive input of the comparator and compare it against a reference on the negative input or another pin via AMUX. You can also optionally use an LPF on the positive input. The output of the comparator can be read in firmware for processing or can be routed out via a pin to the external world.

Figure 1. CMP Block Diagram

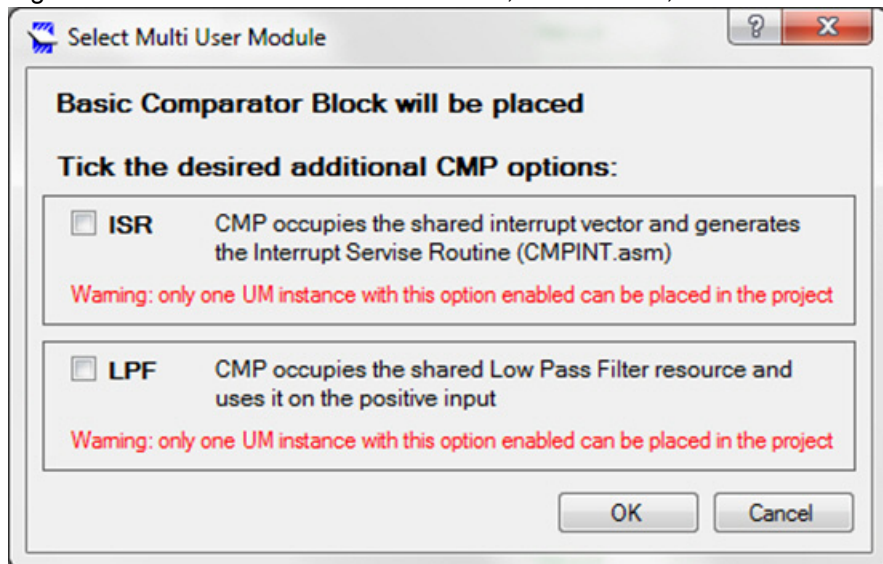


Multi User Module Wizard

The MUM has four combinations for the CY8C24193, CY8C24493, and CY8C24593 devices:

- User module with neither low pass filter (LPF) or interrupt service routine (ISR)
- User module with only LPF
- User module with only ISR
- User module with LPF and ISR

Figure 2. MUM GUI for the CY8C24193, CY8C24493, and CY8C2459 devices

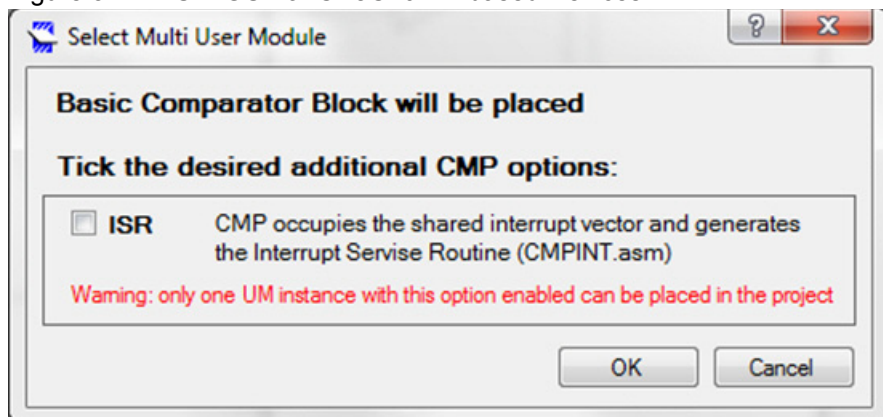


The MUM has two combinations for the CY8C24093, CY8C24293, CY8C24393, and CY8C24693 devices:

- User module without ISR

■ User module with ISR

Figure 3. MUM GUI for CY8C20xx7-based Devices



Parameters and Resources

PMuX

This parameter selects the input to a comparator positive input. The affected registers are CMP_MUX and CS_CR3.

| Type | ENUM |
|---------|---|
| Range | Port_0_1 Port_0_3 AMuX Vref (1.0 V) RefHi (1.2 V) |
| Default | Port_0_1 |

Note If the Vref (1.0 V) or RefHi (1.2 V) is selected – the shared CS_CR3 register is used; therefore, it may override the setting of the user module, which is placed on the adjacent comparator block.

NMuX

This parameter selects the input to a comparator negative input. The affected registers are CMP_MUX and CMP_CR0.

| Type | ENUM |
|---------|--|
| Range | RefHi (1.2 V) RefLo (0.6 V) Vref (1.0 V) AMuX |
| Default | Vref (1.0 V) |

Note If the RefHi or AMuX is selected – the shared CMP_CR0 register is used; therefore, it may override the setting of the user module, which is placed on the adjacent comparator block.

LPF Time Constant

This parameter sets the LPF time constant. The dependency is displayed only for devices based out of CY8C20x66A and only if the comparator with the LPF option is selected in MUM. The affected register is CS_CR3.

| Type | ENUM |
|---------|---|
| Range | 1 μ s 2 μ s 5 μ s 10 μ s |
| Default | 1 μ s |

CMP Out

This parameter connects the comparator output pin. The affected registers are PRTxDR, PRTxDMx, OUT_P1, ADC_TEST2.

| Type | ENUM |
|---------|--|
| Range | Port_1_0 (for CMP0) or Port_1_4 (for CMP1)None |
| Default | None |

Application Programming Interface

The Application Programming Interface (API) routines are given as part of the user module to help you deal with the module at a higher level. This section specifies the interface to each function together with related constants given by the “include” files.

Note

In this, as in all user module APIs, the values of the A and X register may be altered by calling an API function. It is the responsibility of the calling function to preserve the values of A and X before the call if those values are required after the call. This “registers are volatile” policy was selected for efficiency reasons. The C compiler automatically takes care of this requirement. Assembly language programmers must ensure their code observes the policy, too. Though some user module API function may leave A and X unchanged, there is no guarantee they may do so in the future.

For Large Memory Model devices, it is also the caller's responsibility to preserve any value in the CUR_PP, IDX_PP, MVR_PP, and MVW_PP registers. Even though some of these registers may not be modified now, there is no guarantee that will remain the case in future releases.

CMP_Start

Description:

Starts the Comparator by setting the power bit and enables comparator output connection to pin if selected.

C Prototype:

```
void CMP_Start(void);
```

Assembler:

```
lcall CMP_Start
```

Parameters:

None

Return Value:

None

CMP_Stop

Description:

Stops the Comparator by clearing the power bit.

C Prototype:

```
void CMP_Stop(void);
```

Assembler:

```
lcall CMP_Stop
```

Parameters:

None

Return Value:

None

CMP_ChangeMuxes

Description:

Selects the PMuX and NMuX connection for the comparator.

C Prototype:

```
void CMP_ChangeMuxes (BYTE bPMuX, BYTE bNMuX);
```

Assembly:

```
mov A, CMP_PMUX_PORT_0_1 ; bPMuX
mov X, CMP_NMUX_VREF      ; bNMuX
lcall CMP_ChangeMuxes
```

Parameters:

BYTE bPMuX:

| Value Name | Value depends on Placement | | Description |
|------------------|----------------------------|------|-------------|
| | CMP0 | CMP1 | |
| CMP_PMUX_AMUXBUS | 00h | 00h | AMUX |

| Value Name | Value depends on Placement | | Description |
|-------------------|----------------------------|------|---------------|
| | CMP0 | CMP1 | |
| CMP_PMUX_PORT_0_1 | 08h | 80h | P0[1] |
| CMP_PMUX_PORT_0_3 | 0Ch | C0h | P0[1] |
| CMP_PMUX_VREF | 24h | 60h | VREF (1.0 V) |
| CMP_PMUX_REF_HI | 04h | 40h | RefHi (1.2 V) |

BYTE bNMuX:

| Value Name | Value depends on Placement | | Description |
|------------------|----------------------------|------|---------------|
| | CMP0 | CMP1 | |
| CMP_NMUX_AMUXBUS | 42h | 60h | AMUX |
| CMP_NMUX_REF_HI | 02h | 20h | RefHi (1.2 V) |
| CMP_NMUX_REF_LO | 01h | 10h | RefLo (0.6 V) |
| CMP_NMUX_VREF | 00h | 00h | VREF (1.0 V) |

Return Value:

None

CMP_SetLPF

Description:

Selects the LPF constant for the LPF. Available only if the LPF option is selected in MUM.

C Prototype:

```
void CMP_SetLPF (BYTE bLPFVal);
```

Assembler:

```
Mov A, CMP_LPF_1_US ; bLPFVal
lcall CMP_SetLPF
```

Parameters:

BYTE bLPFVal:

| Value Name | Value | Description |
|---------------|-------|-----------------|
| CMP_LPF_1_US | 0h | 1 microsecond |
| CMP_LPF_2_US | 4h | 2 microseconds |
| CMP_LPF_5_US | 8h | 5 microseconds |
| CMP_LPF_10_US | Ch | 10 microseconds |

Return Value:

None

CMP_SetLUT
Description:

Selects the LUT output, which is used for the Comparator output and, therefore, for comparator ISR (if enabled).

C Prototype:

```
void CMP_SetLUT (BYTE bLUTSel);
```

Assembly:

```
Mov A, CMP_LUT_A ; bLUTSel
lcall CMP_SetLUT
```

Parameters:

BYTE bLUTSel:

| Value Name | Value | Description |
|------------------|-------|--------------|
| CMP_LUT_FA | 0h | FALSE (zero) |
| CMP_LUT_A_AND_B | 1h | A and B |
| CMP_LUT_A_AND_NB | 2h | A and ~B |
| CMP_LUT_A | 3h | A |
| CMP_LUT_NA_AND_B | 4h | ~A and B |
| CMP_LUT_B | 5h | B |
| CMP_LUT_A_XOR_B | 6h | A XOR B |
| CMP_LUT_A_OR_B | 7h | A OR B |
| CMP_LUT_A_NOR_B | 8h | ~ (A or B) |
| CMP_LUT_A_NXOR_B | 9h | ~ (A XOR B) |
| CMP_LUT_NB | Ah | ~B |
| CMP_LUT_A_OR_NB | Bh | A or ~B |
| CMP_LUT_NA | Ch | ~A |
| CMP_LUT_NA_OR_B | Dh | ~A or B |
| CMP_LUT_A_NAND_B | Eh | ~ (A and B) |
| CMP_LUT_TRUE | Fh | TRUE (one) |

A > UM's own Comparator output

B > Adjacent (opposite) Comparator output

Return Value:

None

CMP_EnableInt**Description:**

Enables the comparator ISR; this API is not available when the interrupt option is disabled in the MUM selection.

C Prototype:

```
void CMP_EnableInt (void);
```

Assembly:

```
lcall CMP_EnableInt
```

Parameters:

None

Return Value:

None

CMP_DisableInt**Description:**

Disables the comparator ISR; this API is not available when the interrupt option is disabled in the MUM selection.

C Prototype:

```
void CMP_DisableInt (void);
```

Assembly:

```
lcall CMP_DisableInt
```

Parameters:

None

Return Value:

None

CMP_ClearInt**Description:**

Clears the comparator ISR if set; this API is not available when the interrupt option is disabled in the MUM selection.

C Prototype:

```
void CMP_ClearInt (void);
```

Assembly:

```
lcall CMP_ClearInt
```


Parameters:

None

Return Value:

None

CMP_bStatus**Description:**

Returns the comparator output state.

C Prototype:

```
void CMP_bStatus (void);
```

Assembly:

```
lcall CMP_bStatus
jz     .Zero ;if cmp out is low
;else cmp out is high
```

Parameters:

None

Return Value:

Non-zero if the Comparator output is high; otherwise, it is zero.

Sample Firmware Source Code**C Sample Code:**

```
//-----
// CMP sample code
//
// UM should be named "CMP"
//-----

#include <m8c.h>          // part specific constants and macros
#include "PSoC_API.h"    // PSoC API definitions for all User Modules

void main(void)
{
    CMP_ChangeMuxes(CMP_PMUX_PORT_0_1, CMP_NMUX_VREF);
    CMP_Start();
    M8C_EnableGInt;
    while(1);
}
```

ASM Sample Code:

```
;-----
; CMP sample code
;
; UM should be named "CMP"
;-----

include "m8c.inc"        ; part specific constants and macros
```

```
include "memory.inc"      ; Constants & macros for SMM/LMM and Compiler
include "PSoCAPI.inc"     ; PSoC API definitions for all User Modules

export _main

area text (rel,con,rom,code)

_main:
    mov    A, CMP_PMUX_PORT_0_1 ; bPMuX
    mov    X, CMP_NMUX_VREF     ; bNMuX
    lcall  CMP_ChangeMuxes
    lcall  CMP_Start
    M8C_EnableGInt

.mainloop:
    jmp    .mainloop
```

Version History

| Version | Originator | Description |
|---------|------------|-----------------|
| 1.00 | HPHA | Initial version |

Note PSoC Designer 5.1 introduces a Version History in all user module datasheets. This section documents high level descriptions of the differences between the current and previous user module versions.

Document Number: 001-85893 Rev. **

Revised May 15, 2013

Page 10 of 10

Copyright © 2013 Cypress Semiconductor Corporation. The information contained herein is subject to change without notice. Cypress Semiconductor Corporation assumes no responsibility for the use of any circuitry other than circuitry embodied in a Cypress product. Nor does it convey or imply any license under patent or other rights. Cypress products are not warranted nor intended to be used for medical, life support, life saving, critical control or safety applications, unless pursuant to an express written agreement with Cypress. Furthermore, Cypress does not authorize its products for use as critical components in life-support systems where a malfunction or failure may reasonably be expected to result in significant injury to the user. The inclusion of Cypress products in life-support systems application implies that the manufacturer assumes all risk of such use and in doing so indemnifies Cypress against all charges.

PSoC Designer™ and Programmable System-on-Chip™ are trademarks and PSoC® is a registered trademark of Cypress Semiconductor Corp. All other trademarks or registered trademarks referenced herein are property of the respective corporations.

Any Source Code (software and/or firmware) is owned by Cypress Semiconductor Corporation (Cypress) and is protected by and subject to worldwide patent protection (United States and foreign), United States copyright laws and international treaty provisions. Cypress hereby grants to licensee a personal, non-exclusive, non-transferable license to copy, use, modify, create derivative works of, and compile the Cypress Source Code and derivative works for the sole purpose of creating custom software and or firmware in support of licensee product to be used only in conjunction with a Cypress integrated circuit as specified in the applicable agreement. Any reproduction, modification, translation, compilation, or representation of this Source Code except as specified above is prohibited without the express written permission of Cypress.

Disclaimer: CYPRESS MAKES NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS MATERIAL, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. Cypress reserves the right to make changes without further notice to the materials described herein. Cypress does not assume any liability arising out of the application or use of any product or circuit described herein. Cypress does not authorize its products for use as critical components in life-support systems where a malfunction or failure may reasonably be expected to result in significant injury to the user. The inclusion of Cypress' product in a life-support systems application implies that the manufacturer assumes all risk of such use and in doing so indemnifies Cypress against all charges.

Use may be limited by and subject to the applicable Cypress software license agreement.